High-Level Tool Architecture

August 2008
Version 1.0

NIEM Technical Architecture Committee

Record of Changes

No.	Date	Reference: All, Page, Table, Figure, Paragraph	A = Add. M = Mod. D = Del.	Revised By	Change Description
1.0	8/28/2008	All	A	NTAC	Initial draft

Table of Contents

1 Introduction 1
2 Purpose 2
3 Scope 2
4 Strategy 4
5 Principles 5
6 Notation and Terminology 6
7 Data Store Perspective 7
7.1 Reference Business Domain Exchange Model (RefBDEM) Databases 8
7.1.1 Description, Content, and Purpose 8
7.1.2 Search and Retrieval Access 9
7.1.3 Changes to RefBDEMs 9
7.1.4 Export for RefBDEMs 9
7.1.5 Maintenance 9
7.2 NIEM Data Model 10
7.2.1 Description, Content, and Purpose 10
7.2.2 General NIEM Specification 10
7.2.3 Search and Retrieval Access 10
7.2.4 Exports 11
7.2.5 Reference Schema Representations 12
7.2.6 User Interfaces 12
7.2.7 Maintenance 13
7.3 Information Exchange Package Documentation (IEPD) Registry/Repository 13
7.3.1 Description, Content, and Purpose 13
7.3.2 System Interfaces 14
7.3.3 User Interfaces 14
7.3.4 IEPD Specifications 14
7.3.5 IEPD Publish and Subscribe 14
7.3.6 Maintenance 15
7.4 IEPD Implementation Registry 15
7.4.1 Description, Content, and Purpose 15
7.4.2 IEPD Implementation Content Specification 16
7.4.3 IEPD Implementation Search/Retrieval 16
7.4.4 Maintenance 17
7.5 NIEM Component Staging Area (CSA) 17
7.5.1 Description, Content, and Purpose 17
7.5.2 CSA Process Support 18
7.5.3 CSA Search/Retrieval 18
7.5.4 NIEM Change Requests 18
7.5.5 Maintenance 19
7.6 NIEM Configuration Control Tool (NCCT) 20
7.6.1 Description, Content, and Purpose 20
7.6.2 NIEM Configuration Control Tool (NCCT) Issue Access 20
7.6.3 Maintenance 20
7.7 NIEM Knowledge Base 21

7.7.1 Description, Content, and Purpose 21
7.7.2 NIEM Knowledge Base Access 21
7.7.3 Maintenance 21
7.8 Official NIEM Web Site 21
7.8.1 Description, Content, and Purpose 21
7.8.2 Access 21
7.8.3 Maintenance 22
7.9 NIEM Conformance Rule Base 22
7.9.1 Description, Content, and Purpose 22
7.9.2 NIEM NDR Rule Export 22
7.9.3 NIEM-Conformance Validation 22
7.10 Migration Rule Base 22
7.10.1 Description, Content, and Purpose 22
7.10.2 Migration Rule Export 22
8 Information Exchange Package Documentation Life Cycle (IEPDLC) Perspective 23
8.1 Plan Scenario and Analyze Requirements 23
8.1.1 Description, Content, and Purpose 23
8.1.2 Inputs 24
8.1.3 Outputs 24
8.2 Build Exchange Content Model 24
8.2.1 Description, Content, and Purpose 24
8.2.2 Inputs 25
8.2.3 Output 25
8.3 Map NIEM Components 25
8.3.1 Description, Content, and Purpose 25
8.3.2 Inputs 26
8.3.3 Outputs 26
8.3.4 Tools and Interfaces 27
8.4 Build IEPD Artifacts 27
8.4.1 Description, Content, and Purpose 27
8.4.2 Inputs 27
8.4.3 Outputs 28
8.4.4 Tools, Interfaces, and References 29
8.5 Validate IEPD Artifacts 30
8.5.1 Description, Content, and Purpose 30
8.5.2 Inputs 30
8.5.3 Outputs 30
8.5.4 Tools, Interfaces, and References 30
8.6 Document and Assemble IEPD Artifacts 31
8.6.1 Description, Content, and Purpose 31
8.6.2 Inputs 31
8.6.3 Outputs 32
8.6.4 Tools and Interfaces 32
8.7 Publish IEPD 32
8.7.1 Description, Content, and Purpose 32
8.7.2 IEPD Certification and Registration 33

8.7.3 Inputs ... 34
8.7.4 Outputs ... 34
8.8 Implement IEPD ... 34
8.8.1 Description, Content, and Purpose ... 34
8.8.2 Inputs ... 34
8.8.3 Outputs ... 35
8.8.4 Tools and Interfaces ... 35
9 Data Maturity Model Life Cycle (DMMLC) Perspective ... 35
9.1 NIEM Core Subview ... 36
9.1.1 Description, Content, and Purpose ... 36
9.1.2 Inputs ... 37
9.1.3 Outputs ... 37
9.1.4 Tools and Interfaces ... 37
9.2 NIEM Domain Subview ... 38
9.2.1 Description, Content, and Purpose ... 38
9.2.2 Inputs ... 39
9.2.3 Outputs ... 39
Appendix A: List of Interface Specifications ... A-1
Appendix B: NIEM Tool Architecture ... B-1
Appendix C: NIEM Processes, Operations, Functions ... C-1
Appendix D: Tools and Capabilities ... D-1
Appendix E: Acronyms and Abbreviations ... E-1
Appendix F: References ... F-1

Tables

Table 3-1. Information Resources and Processes ... 6
Table 3-2. Specification Types ... 6

1 Introduction

In February 2005, the effort to build the National Information Exchange Model (NIEM) began under the sponsorship of the U.S. Department of Justice (DOJ) and the U.S. Department of Homeland Security (DHS). The initial NIEM design and content was based on, and eventually subsumed, that of its predecessor, the Global Justice XML Data Model (GJXDM). Both programs recognized early that the size and complexity of these models would require software tools to efficiently facilitate widespread adoption and use.

A relatively limited set of GJXDM tools and capabilities have been developed by DOJ as a response to a variety of urgent user needs. These were later adapted and expanded for NIEM. In some cases, tools built for pilot projects were adapted and released rapidly with the understanding that incremental improvements (continued spiral development) could be implemented as user requirements evolved. Use of these tools has been extensive and growing rapidly. However, understanding of the scope, context, and dynamics of the many processes, artifacts, and their associated states during Information Exchange Package Documentation (IEPD) development and NIEM evolution has been lagging because the model was established first to bootstrap IEPD development. It is clear that more tools and capabilities are needed to support the NIEM Program, including information exchange development, model evolution, governance, and management. But more important, NIEM requires a framework that supports tool interoperability through standard interfaces and well-defined artifacts. This will enable the NIEM Program Management Office (PMO) to leverage its limited resources to more rapidly expand automated support for development of NIEM exchanges.

The NIEM Tool Architecture (TA) establishes a strategy and architecture that can efficiently satisfy the need for NIEM-supporting software tools and capabilities. The NIEM TA is a road map and a logical description of NIEM life cycle processes, functions, rules, products, and associated interfaces among them. By publishing open interface specifications, the NIEM Program can economically leverage automated tool support by facilitating interoperability of existing tools and encouraging development of new NIEM-aware tools.

The complete NIEM TA consists of (1) the NIEM High-Level Tool Architecture (HLTA) (this document), which overviews the architecture and identifies all required interface specifications; and (2) a collection of separate documents, each of which is a detailed, open, normative specification identified in the architecture.

The NIEM HLTA (this document) outlines a strategy, principles, life cycle processes, data stores, tools, and capabilities, as well as how these interoperate and assist with NIEM IEPD development, NIEM evolution, governance, and program management. Between the processes and their supporting tools, the NIEM HLTA identifies potential and existing import/export formats and user, system, or maintenance interfaces.

The NIEM TA represents a blueprint[1] for the *to-be* state of the NIEM Program in terms of its processes and support tools.

[1] The NIEM HLTA describes a complete worldview of NIEM for planning purposes and is NOT intended to be an all-or-nothing implementation. The HLTA does NOT contain the normative specifications for the interfaces it identifies. Each specification is prioritized by NIEM governance bodies, individually drafted, evaluated, approved, and subsequently published for implementation and/or use. A specification may be prioritized such that it is never drafted or implemented.

2 Purpose

The NIEM TA is an open plan intended to leverage existing off-the-shelf software tools and facilitate new software tools and value-added capabilities that can support NIEM and its stakeholders.

The early and rapid establishment of useful software tools for NIEM (and GJXDM before) has been a significant factor in the rapid adoption and use of NIEM by an eager user base. While adoption and use are key objectives, the NIEM TA also seeks to accomplish a number of other important objectives, including:

- Maximize information interoperability and reuse
- Minimize the cost of using NIEM (particularly entry costs)
- Reduce time to develop IEPDs
- Minimize the cost of increasing automated support for NIEM
- Leverage more existing and future off-the-shelf tool resources
- Encourage more tool developers to build NIEM-aware tools and capabilities
- Maximize tool integration and interoperation opportunities
- Standardize NIEM processes so they are measurable and repeatable
- Maximize consistency and quality of release products, IEPDs, and associated artifacts
- Minimize NIEM training requirements and technical expertise needed to use NIEM
- Maximize domain, developer, and user self-service

3 Scope

This NIEM HLTA identifies general NIEM specifications as well as specifications for various user, system, and maintenance interfaces and related imports and exports. It locates each specification within the NIEM TA worldview and briefly describes its general functionality and requirements. Subsequently, each specification is normatively defined in a separate document. Together, the HLTA and individual specifications make up the NIEM TA.

The NIEM TA specifies interfaces for tools that support basic NIEM operations, processes, and resources. In particular, the NIEM TA specifies access into the primary NIEM information resources and life cycles. The NIEM HLTA identifies each specification as it relates to the following perspectives:

- A Data Store Perspective. The NIEM Program uses several information resources to store its corporate memory, the model itself, and related products. This perspective identifies interfaces for accessing information and artifacts from these resources.
- The Information Exchange Package Documentation Life Cycle (IEPDLC) Perspective. From a practitioner perspective, the IEPDLC is the primary process for development of the artifacts that define an information exchange specification (an IEPD). The IEPDLC provides a guide to understanding how

IEPDs are built from NIEM and published. It is not intended to be prescriptive. IEPD builders may enter the life cycle at any particular step, as well as adjust the scope of the life cycle to support the level of effort required for their individual IEPD development. This perspective identifies tool interfaces required in building and implementing NIEM IEPDs.

- The Data Maturity Model Life Cycle (DMMLC) Perspective. Over time, NIEM data model stability, quality, and maturity depend on the continuous development and refinement of IEPDs. IEPD development helps to identify new data components, refine existing data components, and identify candidates for harmonization.[2] The IEPDLC and the DMMLC are therefore tightly integrated and, over time, self-optimize their own products (i.e., NIEM and IEPDs). For example, governance committees will review recently developed IEPDs for new components (i.e., extensions created per the NIEM NDR) for potential inclusion into subsequent releases of NIEM. This is one of several ways NIEM evolves to incorporate new data requirements. This perspective outlines interfaces required for the DMMLC to support model maintenance, change management, and governance processes.

While primarily focused on NIEM information resources—the IEPDLC and the DMMLC—NIEM TA scope also includes support for the following operations:

- Program management—Assist PMO to plan, manage, and facilitate the NIEM program.
- Help desk, technical assistance, and training.
- Communications and outreach—Assist NIEM Communications and Outreach Committee (NCOC) to publicize and promote NIEM and inform its stakeholders, as well as other potentially interested communities.

From a user perspective, the NIEM TA addresses the functional tool needs for the following user classes:

- Stakeholders or stakeholder communities with an interest in NIEM.
- Practitioners including IEPD developers and implementers.
- PMO including NCOC, technical assistance staff, content management staff, and development staff.
- Governance bodies including NIEM TAC, NBAC, domain managers and associated committees, and tiger teams.
- Tool developers, including commercial and government developers, contract developers affiliated with the NIEM Program, and development staff.

[2] NIEM harmonization is applicable to a set of data components in a data model, in a single schema, or in multiple schemas. It is the process of modifying the set in an incremental fashion for the purpose of improving the quality of the set with respect to some criteria. For example, one may harmonize to reduce semantic overlap in a set of data components. One may also harmonize a set of data components to ensure uniformity of the vocabulary used in the definitions of those components.

4 Strategy

The NIEM TA specifies the user, system, import/export, and maintenance interfaces that NIEM tools should adhere to in order to interoperate and support NIEM processes. It is not the intent of the NIEM TA to specify functional requirements for tools. Requiring specific functionality constrains tool development and creativity. Furthermore, it can exclude many existing tools that may not contain all functions specified, but nonetheless, might be very useful for particular NIEM processes. Thus, IEPD Life Cycle processes are defined such that developers have some flexibility to use appropriate tools at appropriate steps, and tool developers can report that their tools support specified interfaces and formats.

The NIEM TA strives to leverage good existing and future off-the-shelf software tools independently developed by a variety of sources. To achieve this goal, the NIEM TA ensures that interfaces, imports, and exports between NIEM process steps and functions are open, well-specified formats that use common standards when available and appropriate. This facilitates the adaptation and interoperability of existing tools and establishes specific target formats for future tools, so that each tool may be leveraged at the NIEM process entry or exit point(s) appropriate to its functionality.

The NIEM TA also seeks to reduce cost by promoting and supporting user, developer, and domain self-service for appropriate operational processes. For example, NIEM domains are expected to operate and manage themselves semi-independently; i.e., they (1) govern, harmonize, and manage their own content under the NDR and (2) publish their own namespace updates under the Version Architecture specification and with Program Management Office (PMO) approval. Therefore, tools can assist domain managers with functions such as:

- Automatic enforcement of NDR conformance on updates to domain content.
- Searching NIEM to identify dependencies, duplications, or other components that may require harmonization.
- Generating schemas and other artifacts for updated namespace releases.

Tools can also provide self-service opportunities to IEPD developers through consistent assistance with data requirements analysis, mapping to NIEM, IEPD building operations, IEPD specification conformance, etc.

The NIEM TA also incorporates standard specifications for the storage, search, discovery, sharing, and reuse of various kinds of NIEM data, information, and artifacts. These may include FAQs, lessons learned, NIEM components, reference schemas, IEPDs and their artifacts (schemas, documentation, sample instances, etc.), IEPD implementations, etc.

By establishing open interface specifications, the NIEM TA can promote and facilitate significant increases in automated support for NIEM without requiring corresponding increases in PMO costs or resources.

5 Principles

- Leverage open standards when available and appropriate. This principle increases the potential for interoperability. In some cases, an appropriate standard may not be available; however, it may be possible to use a standard open format (for example, XML Schema) to define a NIEM specification for a particular interface.
- Leverage existing open interfaces and extend or profile as necessary to support NIEM's unique requirements.
- Define platform-neutral specifications. Interfaces should not depend on any particular platform or operating system. This principle maximizes the resources available to support NIEM.
- Expose interfaces through open specifications. This principle ensures complete visibility of the technical detail of each specification.
- Specify bindings for both distributed and in-process implementation scenarios. Interfaces should support both networked and stand-alone applications.
- Define service-level commitments for tools and capabilities when appropriate.
- Focus on NIEM operational processes and support the typical user roles that implement those processes.
- Define formal, open input, and output specifications between appropriate steps or sub-processes of the Information Exchange Package Documentation Life Cycle (IEPDLC) and the Data Maturity Model Life Cycle (DMMLC).
- As appropriate and to the extent possible, identify and specify interfaces that:
    - Facilitate, enhance, or simplify processes.
    - Enable repeatable processes.
    - Allow entrance or exit at any process step.
    - Enable iteration (i.e., bidirectional movement) between life cycle steps or subprocesses when appropriate.
    - Support round-trip engineering so that tools can more easily maintain and synchronize both the XML schemas and the standard artifacts or models that represent them (UML, spreadsheets, etc.).
    - Facilitate reuse of NIEM and its components, as well as IEPDs and their artifacts.
    - Facilitate rapid understanding, detailed search, and easy discovery of NIEM resources including components in the model, IEPDs, documentation, etc.)
    - Enable concurrent development of multiple products that are related by lineage (examples may include extensions of NIEM, related sets of IEPDs, related sets of IEPD artifacts, etc.).

6 Notation and Terminology

This document uses consistent notation to refer to specifications and their corresponding instances. NIEM TA specifications identifiers adhere to the following syntax:

<resource>.<function><spec type code: S, F, U, M, or G>[<disambiguator>] (<name>)

Example: The NIEM spreadsheet specification is referred to as 2.2F2 (NIEM Spreadsheet).

Each specification is identified by the primary information resource or process it is affiliated with. Resources and processes are designated by integers. Table 3-1 lists the integer codes that correspond to these resources and processes. A specification identifier also carries an indication of its functional class (following "."). This is also an integer and is used to group specifications around a function. An integer disambiguator is used to distinguish specifications in the same resource or process with the same general function. (Integer codes do NOT correlate to section numbers.)

Table 3-1. Information Resources and Processes

Code	Information Resources or Processes
1	Reference Business Domain Exchange Models (RefBDEM) or local (nonreference) BDEM
2	NIEM Data Model, its exports, and Naming and Design Rules (NDR)
3	Information Exchange Package Documentation (IEPD) and IEPD migration
4	IEPD implementation
5	Component Staging Area (CSA) and the Data Model Maturity Life Cycle (DMMLC)
6	NIEM Configuration Control Tool (NCCT)
7	NIEM Knowledge Base

Information resources and processes are defined in more detail later in this document.

The five specification types identified in the NIEM TA are listed in Table 3-2. The codes in this table are used in the NIEM TA specification identifiers (<spec type code> in the syntax above).

Table 3-2. Specification Types

Code	Specification Type
S	System interface
U	User interface
F	Import/export or input/output file format
M	Maintenance interface
G	General specification

S—System Interface—This may be a Web Service, an API binding, or a function invocation (for example, executed by a program to manipulate a database or request information from another program, server, or system). This is generally a system-to-system, message- or procedure-oriented communication.

U—User Interface—This may be a stand-alone application interface, a Web application, or a set of static Web pages accessed with a Web browser.

F—Input/Output File Format—A formatted data file (e.g., a database, spreadsheet, XML document, simple ASCII text) that is imported to or exported from a tool, process, database, or repository. Input/outputs may be specified in a variety of formats depending on purpose and whether they must be machine or human readable or both. Classifying file types, formats, purposes, etc. can be relatively complex and is beyond the scope of this HLTA document. The NIEM TA indicates file type(s) where appropriate within each normative specification depending on purpose, requirements, and available standards.

M—Maintenance Interface—This is a user interface that provides typical administrative support functions normally performed by a systems administrator and is specific to an application or product (such as a database). At this time, no maintenance interfaces have been identified that require specifications. This category is defined for potential use in the future.

G—General Specification—This is a normative specification that defines rules, principles, configurations, and/or other standard regulating constraints for aspects of the NIEM Program.

The NIEM TA refers to the term *reference tools*. The PMO designates and makes available reference tools as the baseline set against which other tools are measured for conformance and correctness. Characteristics of reference tools include:

- PMO-sponsored (not necessarily PMO-developed).
- Freely available to facilitate adoption of new releases.
- Generally available with or shortly after a new release.
- Support changes in structure and content based on release requirements (e.g., additional metadata, search capabilities).
- Serve as an authoritative reference implementation.
- Serve as a means for validating/testing interfaces and interoperability of other independently developed tools.

7 Data Store Perspective

NIEM uses the following databases, registries, and repositories to hold its constituent information resources:

1. Reference Business Domain Exchange Models (RefBDEM)
2. NIEM Data Model
3. Information Exchange Package Documentation (IEPD) Registry/Repository
4. IEPD Implementation Registry
5. NIEM Component Staging Area (CSA)
6. NIEM Configuration Control Tool (NCCT)

7. NIEM Knowledge Base
8. http://www.niem.gov
9. NIEM Conformance Rule Base
10. Migration Rule Base

A high-level description of the purpose and/or functions, content, and interfaces for each current and potential resource follows.

NIEM is considered a public standard and therefore most, but not necessarily all, of these data stores will be exposed through public Web sites. Therefore, all will have minimum security requirements typical of public Web sites. However, some data stores may also have additional, possibly unique security requirements that depend on the sensitivity of the material stored, how it is accessed, who may access it, and the risks of unauthorized release. Such requirements may include special user authentication, password protection, data encryption, etc. Since for each data store the security requirements may differ, these will be defined in the associated NIEM TA specifications.

7.1 Reference Business Domain Exchange Model (RefBDEM) Databases

7.1.1 Description, Content, and Purpose

An information exchange is defined in terms of its business context. Examples of business context include, but are not limited to, sending agencies, receiving agencies, events that trigger data exchange, conditions of the exchange, process, process flow, and data requirements. An agency models its exchanges by building a Business Domain Exchange Model (BDEM) database that contains such business context. Within a given domain, existing BDEMs from various agencies can be analyzed for commonalities. From these BDEM commonalities a Reference BDEM (RefBDEM) database can be bootstrapped, normalized for, and harmonized over the business domain it represents. This implies the following potentially useful standardizations:

- One RefBDEM exists for each business domain.
- A RefBDEM and the BDEMs from which it was built have identical database structure. Note this would be true within a given domain. This does NOT imply that RefBDEMs from different domains necessarily have identical structure.
- A RefBDEM and the BDEMs from which it was built may all have different, but possibly overlapping, content.
- For a given domain, the same interfaces can be used to access information from RefBDEMs and BDEMs.

Once a RefBDEM database is available for a given domain, an agency in that domain can use it as a template or starting point (either partially or entirely) to develop a Business Domain Exchange Model (BDEM) database to represent its own local exchange environment. Obviously, RefBDEMs are important artifacts that can and should be shared, reused, or adapted within a particular domain.

As an example, suppose a state criminal justice agency wants to model its arrest process. Then, it may start with a Law Enforcement RefBDEM database that contains boilerplates for typical arrest-related information exchanges described in terms of standard business context for the Law Enforcement domain.

7.1.2 Search and Retrieval Access

The following interfaces provide search and retrieval access to RefBDEMs:

1.1S1 (RefBDEM SI): A system interface for online search/retrieval access to a RefBDEM or BDEM. This system interface defines both how to search and what entities can be searched (query) as well as expected results (response). For example: A tool might consult the Law Enforcement RefBDEM for all exchanges associated with an Arrest event or process (business context). Find all Law Enforcement business domain exchanges, where process = Arrest.

1.1S2 (RefBDEM API): An application programming interface (API) for baseline search, retrieve, and edit capabilities that can be used to manipulate a RefBDEM or BDEM export per 1.3F (RefBDEM Export) below.

7.1.3 Changes to RefBDEMs

1.2F (RefBDEM Change Request): A change request format for a RefBDEM. A RefBDEM is a reference model for an entire domain. As such, changes to a RefBDEM should be controlled and applied through a formal evaluation process.

7.1.4 Export for RefBDEMs

1.3F (RefBDEM Export): A file format that a tool can import to directly access a RefBDEM or manipulate a BDEM. This specification should allow for partitioning of a model by its business context types. For example, it should be easy to isolate and/or extract and export just the data model, just the sending agencies, or just the receiving agencies portions of the RefBDEM (or BDEM).

7.1.5 Maintenance

A formal specification for a RefBDEM maintenance interface is unnecessary. Since RefBDEMs represent reference models for entire domains, RefBDEM updates must be evaluated and controlled by domain governance processes using 1.2F (RefBDEM Change Request) above. Automatic application of updates to RefBDEMs from user submissions is unlikely; updates to RefBDEMs will generally require manual application.

7.2 NIEM Data Model

7.2.1 Description, Content, and Purpose

The NIEM Data Model comprises Core, domains, code lists, adapters for external standards components, and support structures. It is a data store that contains types, properties, relationships, metadata, semantics, and structural characteristics of the NIEM data components used to generate XML schemas for building IEPDs. This resource provides a NIEM authoritative source and facilitates search and discovery of NIEM components, associated metadata about them, and relationships between them. Metadata includes component names, namespaces, types, definitions, key words, use examples and descriptions, code value sets, and classification information. In addition to XML schemas, other artifacts are generated from this database (including documentation, metadata, and other representations). The NIEM TA also supports federated access to the NIEM Data Model components and metadata.

Effective search and discovery of NIEM components and associated metadata (including appropriate context) are absolutely critical to the efficiency of both IEPD development and NIEM governance processes. Finding the most appropriate NIEM component(s) quickly is needed to map, model, build a schema, evaluate a release, and a host of other NIEM activities. Improving search is a function of several related factors, including the design of the metadata contained in the model, the quality of that metadata, the availability or access to the model and its metadata, and the methods that tools apply to search the model. The NIEM HLTA introduces specifications for the design and availability/access of the metadata. However, metadata quality is a result of good governance of the model, and search methods are dependent on how the tools use the available metadata (with the understanding that good metadata design and quality enable better methods for tools to apply).

NOTE: The NIEM Data Model and its XML Schema rendering (the release reference schemas as specified in 2.2F1 (NIEM Ref Schemas)) are not the same. The *Structures* and *Appinfo* schemas are added to the release after it has been generated from the model. *Structures* and *Appinfo* are used to support the XML Schema representation of the model only and are not part of the model itself. This separation of model and schema representation has enabled efficient and cost-effective implementations of some fairly significant mechanisms (such as associations, roles, and augmentation) without requiring drastic changes to the basic framework or support tools. If required, a system interface and/or file export could be defined to provide access to the complete release (that includes *Structures* and *Appinfo* components).

7.2.2 General NIEM Specification

> 2.1 (NIEM NDR): [13] The NIEM Naming and Design Rules (NDR) specification. The NDR is a normative document that defines naming and structure for all XML Schema components in the NIEM reference schemas (releases).

7.2.3 Search and Retrieval Access

The following interfaces provide search and retrieval access to the NIEM Data Model:

2.3.1S (NIEM Search SI): A system interface for search and retrieval within the NIEM Data Model. Typical functions to access NIEM components, relationships, or metadata might include:

getPropertyByName(<name>)
getTypeByName(<name>)
getBaseType(<type>)
getParentTypes(<type>)
getSubstitutionGroup(<property>)
getSubstitutableProperties(<property>)
getDefinition(<property>)
getAllAssociations(<type>)

These functions are NIEM-aware and specific. As such, they are tailored for NIEM IEPD development and therefore are designed to provide capabilities that are beyond the scope of general federated search.

2.3.2S (NIEM Federated Search SI): A metadata model and associated system interface for standards-based, federated, read-only, search/retrieval access to NIEM components, artifacts (IEPDs), and metadata. NIEM can leverage the Common Terrorism Information Sharing Standards (CTISS) Federated Registry project, which specifies federated access to NIEM components, IEPDs, and metadata. This specification includes both a metadata model based on ebXML Registry Information Model [F-12] and a system interface based on ebXML Registry Services [3].

See also the IEPD Registry/Repository section of this document, which uses this specification for IEPD access. Note that this federated service is much more generic than 2.3.1S (NIEM Search SI) and is not intended to provide more capabilities than federated search and discovery of NIEM components and artifacts.

2.3.3S (NIEM Export DB API): An API for search and retrieval functions to be used with 2.2F3 (NIEM Access Export DB) (described below). Functions supported by this API are similar to interfaces defined by 2.3.1S (NIEM Search SI).

7.2.4 Exports

The following exports for the NIEM Data Model are defined:

2.2F2 (NIEM Spreadsheet): [15] A human readable rendering of a given release of the NIEM data model in a tabbed spreadsheet (Microsoft Excel); hyperlinked for efficient navigation.

2.2F3 (NIEM Access Export DB): [4] A database rendering of a given release of the NIEM Data Model in Microsoft Access format.

2.2F4 (NIEM Spreadsheet Export DB): [16] Relational database tables for a given release of the NIEM Data Model rendered in Microsoft Excel format; each tab represents one database table.

2.2F5 (NIEM CSV Export Files): [7] Relational database tables for a given release of the NIEM Data Model rendered in Comma Separated Value (CSV) file format.

7.2.5 Reference Schema Representations

2.2F1 (NIEM Ref Schemas): [14] A set of XML schemas representing a single coherent release of the NIEM Data Model, including NIEM Core, domains, code lists, external standards and associated adapters, and other support schemas such as *AppInfo* and *Structures*.

2.2F6 (NIEM XML Export File): A set of XML schemas that contain all NIEM components and associated metadata for a given release of NIEM, including components and metadata in *Structures* and *Appinfo*. This is a non-normative form (i.e., it cannot be used to validate IEPDs) of a NIEM release intended as an easily parsed format for all NIEM XML Schema components and metadata.

7.2.6 User Interfaces

2.2U (NIEM URI Pages): [17] URI pages are static Internet referenceable XHTML documentation pages for NIEM properties and types. Tools and documents can link to or retrieve the latest page for a NIEM component by simply referencing its URI (Uniform Resource Identifier). Each NIEM namespace URI is extended (suffixed) with a component name from that namespace. The URI for each NIEM component and the URL for its associated documentation page are equivalent. So the component documentation page can be accessed through its assigned URI. Example: the URI page (and corresponding URL) for NIEM 2.0 “ActivityType” is http://niem.gov/niem/niem-core/2.0/element/ActivityType

7.2.7 Maintenance

A formal specification for maintenance and direct manipulation of the authoritative copy of the NIEM Data Model is not specified at this time. Updates to the model are controlled by and depend on NIEM governance processes. Updates are generally manual because of complexity. Refer to the Component Staging Area (CSA) section in this document for discussion of NIEM change requests.

7.3 Information Exchange Package Documentation (IEPD) Registry/Repository

7.3.1 Description, Content, and Purpose

IEPDs comprise several XML schemas that define a consistent set of XML data exchange instances. Since the development of a single IEPD can require significant time and effort, leveraging previous similar work by sharing, reuse, and adaptation of IEPDs is highly desirable. Therefore, the NIEM TA strongly supports and encourages IEPD sharing and proliferation of registries by employing several key concepts:

1. Establish standards for federated search and discovery so that searching one registry searches all registries that implement the federation standards. This enhances sharing by saving the time to both find and search individual registries.

2. Specify that IEPDs are self-documenting. A complete IEPD contains documentation needed to understand its business requirements, business context, points of contact, hardware- and software-independent aspects of implementation, etc.

3. Specify that IEPDs are self-contained. A complete IEPD contains a structured catalog of its schemas, metadata, and documentation that normatively describes the purpose and nature of each file. This has a number of advantages. For example, it will enable automated scanning and processing of registered IEPDs. Extension schemas can be identified and their XML components harvested for consideration as new NIEM candidate components.

4. Specify that IEPDs are self-describing. A complete IEPD contains a structured file that describes all metadata required for registration and federation. This enables IEPDs to be registered anywhere,

5. Specify that each IEPD is assigned a URI. This ensures that each IEPD can be positively distinguished from others as well as from other versions of itself. The URI can also be a URL that resolves directly to the author's master copy of the IEPD. This further facilitates harvesting of IEPDs and their artifacts, analysis of component usage, collection of statistics for PMO use, etc.

By hosting an IEPD registry and repository accessible from http://niem.gov, PMO can harvest and maintain copies of IEPDs to support program management and governance.

7.3.2 System Interfaces

For federated search and discovery, see 2.3.2S (NIEM Federated Search SI), defined under NIEM Data Model system interfaces.

7.3.3 User Interfaces

User interfaces for PMO-provided Web applications are standard Web browser interfaces.

7.3.4 IEPD Specifications

3.1.1 (IEPD Specification): [11] Specifies the format and content of an IEPD as an archive file set. Defines basic metadata for registry purposes. An IEPD contains a number of artifacts (mandatory and optional), including schemas, documentation, and sample instances, as well as both 3.2.3F (IEPD Metadata) and 3.2.4F (IEPD Catalog) files. Defines an IEPD archive as independent, self-documenting, self-contained, and self-describing; therefore, it may be registered in multiple locations.

3.2.3F (IEPD Metadata): [10] metadata.xml—An XML file that validates against an associated XML schema and contains standard required and optional metadata for an IEPD archive. Content of this file is used for search and discovery of the IEPD in IEPD registries.

3.2.4F (IEPD Catalog): [8] catalog.xhtml—An IEPD manifest file listing the file name, descriptions, nature (type), and purpose (use) of all files within an IEPD archive (including itself); an XHTML file type in Resource Directory Description Language v2.0 [22] format. The purpose of this file is to enable automatic identification and processing of the files in an IEPD archive.

7.3.5 IEPD Publish and Subscribe

3.1.3F (IEPD Submission): A specification for submitting an IEPD to registry/repositories. This could simply be a service that sends, or an e-mail address to which is sent, the IEPD. Specific requirements and metadata must be identified. Note that the same specification could also be used for submission of the IEPD to the NIEM PMO or another authority for approval, validation, and certification if required by a registry/repository owner. See also 3.1.6S (IEPD Certification Request SI).

3.1.5F (IEPD Subscription): A specification for subscription data from users wishing to receive IEPD update information and notifications. 2.3.2S (NIEM Federated Search SI) can be used with this data to execute a federated search for each subscriber. A common standard subscription specification based on 3.1.1 (IEPD Specification) and 2.3.2S (NIEM Federated Search SI) will make it easier for subscribers to receive update notifications from multiple IEPD registries and will promote a federated approach to IEPD registries and repositories (expected to proliferate). Requirements should be relatively simple and include logging and tracking of subscriber e-mail addresses and associated search criteria metadata. IEPD subscriptions may be leveraged for other NIEM notifications, including release announcements, distributions, model updates, change impact reports, etc. The self-describing nature of 3.1.1 (IEPD Specification) ensures that IEPDs contain their own metadata for discovery purposes.

Example: A given individual or organization wants to be notified of published changes to existing IEPDs in the Justice domain.

- Entity submits 3.1.5F (IEPD Subscription) to NIEM IEPD Registry/Repository containing return e-mail address and metadata describing criteria for notification (in this case, domain="Justice"; other federated search attributes are possible)
- IEPD Registry logs this subscription with e-mail address and criteria.
- Periodically, an agent executes a 2.3.2S (NIEM Federated Search SI) of IEPD registries for each subscription and its corresponding criteria.
- Agent e-mails results to addresses provided in subscriptions.

7.3.6 Maintenance

Maintenance functions are registry-specific and, therefore, outside scope.

7.4 IEPD Implementation Registry

7.4.1 Description, Content, and Purpose

An IEPD Implementation Registry will store system design details from IEPD implementations provided at the discretion of the implementing organization. Details may include hardware/software configurations and specifications. Depending on the scope and scale of such efforts, the products, results, and lessons learned could be extremely valuable to others who are executing or planning similar efforts. This registry may have several forms and various kinds of content (subject to legal constraints and considerations):

- Simple Web site that references (links to) implementer sites.
- Metadata summary describing the implementation and what it supports.
- Detailed system configurations, specification, diagrams, etc.
- Evaluated product lists (used in IEPD implementations).

Example: An agency's record or case management system may be implemented to feed data into N-DEx via the N-DEx IEPD. Other agencies have similar requirements for N-DEx submissions. Commercial products are often designed to support such common interfaces. Therefore, an IEPD implementation registry may share information about implementing the N-DEx IEPD, such as standard commercially or openly available platforms, user interfaces, database applications, and plug-ins, already adapted to exchanges using this IEPD.

7.4.2 IEPD Implementation Content Specification

4.1F (IEPD Implementation): Defines metadata, files, organization, and formats for an IEPD Implementation package that can be submitted to the registry. Package may be similar to 3.1.1 (IEPD Specification) so that approved data may be auto-processed into the registry. Specification may include the following artifact files:

Metadata file similar to 3.2.3F (IEPD Metadata) for an IEPD
Catalog file similar to 3.2.4F (IEPD Catalog) for an IEPD.

IEPD implementation files may include:
- System design documentation
- System-specific requirements
- Business rules
- Exposed APIs, Web Services, or other system interfaces
- Hardware/software products used or adapted
- Hardware/software configurations
- Test data and results
- Statistics: cost, constraints, transaction volume, timing, quality of service, etc.
- General lessons learned

7.4.3 IEPD Implementation Search/Retrieval

4.1S (IEPD Implementation SI): A system interface that enables tools to search/retrieve metadata and relationships from an IEPD Implementation Registry. Typical functions may include:
- getImplementationsByIEPDName(<name>)
- getImplementationsByOrganization(<organization>)
- getImplementationsByVendor(<vendor>)
- getImplementationsByVendorProduct(<vendor-product>)

4.1U (IEPD Implementation WebApp): A Web application that enables online search/retrieval access to an IEPD Implementation Registry. Example: Search for all Record Management System implementations of the N-DEx 1.0 IEPD. Return appropriate packages formatted per 4.1F (IEPD Implementation).

7.4.4 Maintenance

Maintenance is based on 4.1F (IEPD Implementation). Auto-update with submission packages is unlikely. Submission packages must be reviewed before publishing to registry. 4.1F (IEPD Implementation) can be auto-processed after approval.

7.5 NIEM Component Staging Area (CSA)

7.5.1 Description, Content, and Purpose

Figure 15, page 42 of the NIEM Concept of Operations [6] document is a high-level view of the harmonization process within the Data Model Maturity Model Life Cycle (DMMLC). An important resource depicted in this figure is the NIEM Component Staging Area (CSA), a dynamic collaborative workspace with associated processing capabilities that supports governance activities and the DMMLC. This facility is logically partitioned into workspaces for Core and each domain (and others as needed) and is designed to support a self-service and independent management approach to governance tasks and operations. Five potential classes of functionality include:

- Capabilities for preprocessing artifacts or components prior to staging in the CSA for subsequent governance committee action. Examples: identification of semantic overlaps among staged components or components across Core and domains; harvesting candidate components from extension schemas or current domain namespaces; NIEM-conformance validation and documentation validation; common tools for searching, parsing, and translation as required.
- Well-organized areas for preparing, staging, searching, and viewing components under consideration. These areas have consistent views for various types of components (simple, complex, Associations, Roles, Type Augmentations, codes and code lists, substitution groups, external standards, etc.). Component views may include context such as documentation (definitions and metadata), IEPD artifacts, or other artifacts that contain components. Graphical tree views of components may be useful.
- Capabilities for semiauto analysis and generation of statistics for various types of change impacts. Examples: identification of dependencies; how many (and which) IEPDs, domains, or other components are affected by a change; version architecture analysis and application of rules.
- Discussion boards to facilitate and record open collaboration among subject-matter experts during modeling, harmonization, and review activities; capability to be notified of changes made to the CSA.
- Capabilities that support resulting approved releases and associated products for both the NIEM PMO (for Core) and domain administrators (for domains).

7.5.2 CSA Process Support

5.1U (CSA WebApp): A Web application that supports NIEM Core and NIEM domain governance processes. This WebApp should provide capabilities such as:

a. Search, retrieve, and browse CSA content. Examples: Search for all
   (1) domain candidate components dependent on one or more domains.
   (2) domains that depend on (i.e., use) a given component.
b. Online collaboration and discussion to facilitate harmonization.
c. Public content publication (e.g., major, minor, micro, and update releases).
d. Private content publication (e.g., suggested remodeled components, metadata changes).
e. Voting to support decisions.
f. Change notifications.

7.5.3 CSA Search/Retrieval

5.1S (CSA SI): A system interface for search/retrieval and access to the Component Staging Area (CSA) for commenting. This specification will depend on 5.1U (CSA WebApp) functional requirements discussed below. It is likely a low-priority interface, at least until 5.1U (CSA WebApp) has been specified, implemented, and put into practice.

7.5.4 NIEM Change Requests

The current change process for NIEM relies on the NCCT, governance, and the Component Mapping Template (CMT); eventually this will include 5.3G (Version Architecture). CMT was designed and used for batching large amounts of content from domains for integration into NIEM 1.0 and 2.0. The CMT has been a simple, effective, offline tool for quickly collecting potential NIEM content from a domain, reviewing it for consistency, and rapidly integrating it into the model. It has been used to define:

- Mappings for review with subject-matter experts
- Requirements for building or refining IEPD schemas
- Requirements for developing IEPs
- Requirements for data component candidates for NIEM additions or corrections

Attempts to expand its scope into a multipurpose tool have uncovered a few weaknesses:

- Needs a more precise notation for mapping components (e.g., XPath).
- Needs more context information for mapping (parent, child, containing structure, authority).
- May not be capable of handling multiple requirements (above) effectively.

Potential approaches for improving the CMT may be:

- Redesign content and format of the CMT to maximize effectiveness for all requirements.
- Specify alternative input formats for each requirement.
- Hybrid forms of above.

The following specifications replace (or subsume) the CMT. These specifications would be used to submit NIEM changes to the CSA for subsequent processing and review by appropriate governing bodies:

5.2F (CSA Change Request): A standard NIEM update request specification and format that covers requests for new components or changes to existing components for Core, a domain, a code list, or any combination of these. This specification is used for several kinds of NIEM update requests for both domain and Core updates gathered from NCCT and IEPD extension schemas. It incorporates the following requirements/functionality:

Definition of NIEM NDR-conforming components with metadata and context.
Specification of components to be changed, remodeled, or refactored.
Specification of where components should be inserted or integrated.
Specification of code lists or changes to existing code lists.
Specification of external standards.
Inclusion of clarifying explanations or comments as necessary.
Inclusion of metadata to justify as new NIEM data requirements or changes.
Inclusion of submitter metadata: name, authority, organization, phone, e-mail,...
Capability to auto-process requests into CSA and display to governing bodies.
Capability to auto-aggregate multiple requests into one.
Capability to auto-process and integrate final approved packages into NIEM.

5.2S (CSA Change Request SI): A system interface for submitting requests for NIEM Core or NIEM domain changes 5.2F (CSA Change Request) to the CSA. This interface may also be supported through an e-mail submission address with 5.2F (CSA Change Request) attached.

7.5.5 Maintenance

Automatic updates using update request submissions are unlikely. Governance, harmonization processes, and constraints on versioning will limit the degree of automation possible upon receipt of a change request. All Core and domain updates must be initiated, processed, reviewed, refined, and finalized by governing committees in the CSA. At this point, it may be possible to

automate approved update requests to process changes to NIEM for subsequent releases. 5.3G (Version Architecture) will further detail procedures.[3]

7.6 NIEM Configuration Control Tool (NCCT)

7.6.1 Description, Content, and Purpose

This issue-tracking system records and maintains all NIEM issues related to errors, new content requests, content changes, requirements, architecture, and so forth that NIEM governance bodies must review, discuss, and resolve. Each NIEM release change log references the NCCT issue(s) that captures the discussion, rationale, and decision for (or against) a change. The NCCT is partitioned into logical discussion groups based on the NIEM architecture.

The NCCT is a nonpublic, account-based tracking system. Only NIEM PMO, staff, governance committee members, and beta testers have accounts. All accounts have read access to all entries. Update permission is granted to the members of a governance committee associated with the appropriate NCCT discussion group(s).

7.6.2 NIEM Configuration Control Tool (NCCT) Issue Access

6.1U (NCCT WebApp): Web application. Functionality is based on open-source Bugzilla [1] software, and includes:

- Search by any combination of the attributes that define an issue.
- Full-text search for issue entries (includes wildcards).
- Enter comments, responses to comments, etc.
- Attachment files (various formats) to support issue discussions.
- URI/URL reference to each NCCT issue.
  - Example: http://niem.gtri.gatech.edu/ncct/show_bug.cgi?id=123
- Exports of issue data, including:
  - Short-format Comma Separated Value file (CSV)
  - Long-format Web pages (HTML)

7.6.3 Maintenance

Bugzilla-specified functionality for typical system administration operations:

- Manage user accounts.
- Create new or modify existing discussion groups.
- Move issues between discussion groups.
- Set various configuration parameters.

[3] An exposed interface between the CSA and the NIEM Data Model for updating the data model with approved changes coming from the CSA cannot be determined as yet. As implied above, model updates are dependent on versioning architecture (still in development) and are tightly coupled to governance processes. Furthermore, it is generally desirable to allow model updates only through a single private interface. If cross-domain dependencies can be controlled, it may be possible to allow an administrator to update his/her domain data model independently, thus promoting the self-service concept. However, updates to the Core data model will certainly be private since all other namespaces depend on Core, which will not be updated as often as domains.

7.7 NIEM Knowledge Base

7.7.1 Description, Content, and Purpose

The NIEM Knowledge Base [12] is maintained by the NIEM help desk. This database organizes and stores all questions, answers, suggestions, etc. submitted to the help desk by the public via e-mail or telephone. Many submissions also become or generate entries to NCCT. The NIEM Knowledge Base content is publicly available on the Internet by browser.

7.7.2 NIEM Knowledge Base Access

7.1S (NIEM KB SI): A system interface that enables external tools or servers to search/retrieve entries from the NIEM Knowledge Base [12].

7.1U (NIEM KB WebApp): A URI/URL that can be assigned to each knowledge record (similar to NCCT). The NIEM Knowledge Base [12] already provides a standard browser-based Web application interface that includes:

- Search, retrieve, and browse functions.
- Ability to submit questions and feedback.
- User workspace ("My Stuff") for checking status of IEPD submissions.

7.1F (NIEM KB Export): Define an NIEM Knowledge Base export format. This could be useful to independent training programs and other tools.

7.7.3 Maintenance

An open maintenance interface for the NIEM Knowledge Base is outside the scope of the NIEM TA. This interface simply executes application-specific user account and database maintenance functions.

7.8 Official NIEM Web Site

7.8.1 Description, Content, and Purpose

This resource (http://niem.gov) is the official PMO public Web site for all public information about NIEM, including announcements, calendars, training, conferences, downloads, tools, etc. Some information is content on this Web site (e.g., NIEM release distributions, key documentation, announcements, URI pages, newsletters). Other information is referenced from this Web site (e.g., the NIEM Knowledge Base, the OJP IEPD Clearinghouse, NIEM-aware tools).

7.8.2 Access

Search/retrieval/browse/download access is through a standard Web browser interface.

7.8.3 Maintenance

An open maintenance interface to the NIEM Web site is outside the scope of this document. This site is PMO-approved and contractor-controlled. It is maintained on a standard Web server using any off-the-shelf Web content management system.

7.9 NIEM Conformance Rule Base

7.9.1 Description, Content, and Purpose

The NIEM conformance rule base contains formal rules that correspond to 2.1 (NIEM NDR). This is a normative set of rules used to auto-validate that a schema conforms to the NIEM NDR. These rules generally employ a combination of formats, including XML Schematron and eXtensible Stylesheet Language Translation (XSLT) and require a procedural language such as Java to process. Note that Schematron and XSLT do have limits—not all NDR rules can be codified using these languages. Some rules may require another language or other tools to auto-check. Furthermore, some rules are very subjective and can only be checked by human inspection and evaluation. The NIEM TA intent is to facilitate tools and capabilities that can automatically validate as many of the NDR rules as possible.

7.9.2 NIEM NDR Rule Export

> 2.1F2 (NIEM Conformance Rules): A formal representation of the English rules specified by the NIEM NDR. This file is used to auto-validate that a schema is NIEM-conforming (i.e., conforms to the NIEM NDR).

7.9.3 NIEM-Conformance Validation

See 3.1.2S (NIEM Validation SI).

7.10 Migration Rule Base

7.10.1 Description, Content, and Purpose

This rule base is employed to transform a 3.2.1F (NIEM Wantlist) for a NIEM schema subset generated from an older version of NIEM into a *wantlist* for the next major release version. The rule base stores knowledge of component changes from one major release to the next. Current rules address changes from GJXDM 3.0.3 to NIEM 2.0 and from NIEM 1.0 to NIEM 2.0.

7.10.2 Migration Rule Export

> 3.4F (NIEM Migration Rules): Defines an export file containing the Migration Rule Base specified in XML format that validates against an XML schema.

8 Information Exchange Package Documentation Life Cycle (IEPDLC) Perspective

The NIEM Information Exchange Package Documentation Life Cycle (IEPDLC) consists of a set of sequential processes or steps for planning, developing, and implementing Information Exchange Package Documentation (IEPDs), and subsequently transmitting Information Exchange Packages (IEPs, i.e., instances of IEPDs). The IEPDLC is a reference framework that describes flexible but repeatable procedures for implementing information exchanges in NIEM. In this document, the IEPDLC framework is used to identify interfaces, inputs, outputs, and associated specifications that can facilitate automated tool support.

The IEPDLC processes are:

1. Plan Scenarios and Analyze Requirements
2. Build Exchange Content Model
3. Map NIEM Components
4. Build IEPD Artifacts
5. Validate IEPD Artifacts
6. Document and Assemble IEPD Artifacts
7. Publish IEPD
8. Implement IEPD

What follows is a summary description of each process, its interfaces (inputs and outputs), and its internal subprocesses that are useful as exposed interfaces. Requirements for proposed or existing specifications are identified where appropriate. Some steps in the IEPDLC will interface with previously described NIEM database and registry/repository resources. In such cases, the specification identified for these resources will be referenced for use in those steps.

The inputs and outputs identified are generally minimal sets. Depending on the nature of the information exchange being built, there may be other useful inputs or outputs that an IEPD developer uses or generates. Though optional, these additional inputs and outputs can and should be carried forward for inclusion in the final IEPD, especially if they provide additional context that is helpful to understanding the planning, development, implementation, or use of the IEPD.

Note that all outputs can be carried through the life cycle to subsequent steps as part of the IEPD. However, only those actually used in a given step are mentioned as its inputs. The final draft IEPD contains all accumulated required outputs from the life cycle, as well as optional outputs that are useful.

8.1 Plan Scenario and Analyze Requirements

8.1.1 Description, Content, and Purpose

The initial step in the IEPDLC uses scenario planning to develop use cases and analyze requirements for information exchanges, both current and future. The ultimate objective of this step is to identify the various information transmissions that must take place in a given domain.

8.1.2 Inputs

Inputs include scenario-related items as follows:

- Formal and informal samples of records (both paper and electronic), including correspondence, mail, forms, orders, directives, status reports, transactions, dispositions, memorandums, etc.—any artifact of information or data communication relevant to the scenario.
- Policy and procedure manuals, mission statements.
- Communication lists—mail, e-mail, telephone, fax, etc.
- Existing relevant scenarios or exchange models.

These inputs are used to determine the nature and business context for each exchange required in the scenario and to define these exchanges in a uniform format. The first three inputs are diverse and follow a variety of formats. The fourth input adheres to 1.3F (RefBDEM Export) format. This motivates a RefBDEM repository and potentially BDEM sharing. As more BDEMs are built, and in turn, RefBDEMs are synthesized from them, a RefBDEM (or BDEM) repository can be searched using system interface 1.1S1 (RefBDEM SI) for exchange models relevant to the scenario. Once found, the model(s) can be exported in 1.3F (RefBDEM Export) format and accessed locally to extract a specific exchange or data using 1.1S2 (RefBDEM API).

8.1.3 Outputs

Outputs for this process are information exchange descriptions (RefBDEMs and BDEMs) defined by 1.3F (RefBDEM Export) containing business context such as:

- Sending and receiving entities (and/or types of).
- Relevant business processes during which transmissions are made.
- Events that trigger transmissions.
- Conditions under which the transmissions are executed.
- Description of the data transmitted:
  - Name, type, and definition
  - Constraints (cardinality, patterns, lengths, validity dates, etc.)
  - Other metadata as required

8.2 Build Exchange Content Model

8.2.1 Description, Content, and Purpose

The previous step (Analyze Requirements) documents the kinds of data transmissions required and their context; that is, things that are important to understanding their nature. This step constructs a conceptual model to describe the data entities, their semantics, and the relationships among them. This conceptual model is henceforth referred to as an *exchange content model*. There are several ways to represent an exchange content model. One popular graphical representation is the Universal Modeling Language (UML). Regardless of the format used, data requirements are usually represented in some visual format (such as UML) that defines and illustrates how data is organized into information objects to achieve intended business communications objectives. At this stage, the exchange content model may incorporate NIEM component names and/or structures, but it is not required to do so. Relevant preexisting

exchange content models may be used in whole or part, or may be adapted to build this model. Other business context that defines the exchange (e.g., senders, receivers, processes, events, and conditions) is also carried forward to this step and will become part of the final IEPD.

The objective of this step is to ensure that the correct data entities, semantics, and relationships are defined conceptually. Exactly how that is done is NOT prescribed by NIEM processes. A UML exchange content model is not required. However, advantages to use of UML include its popularity and its standardization as both a graphic and a serialized format. Its serialized format, XML Metadata Interchange (XMI), allows many software tools to automatically interpret and process a UML model.

8.2.2 Inputs

- Data descriptions from the relevant RefBDEM or BDEM (the data subset of its 1.3F (RefBDEM Export)).
- Existing exchange content models (may be in a variety of formats) that are relevant and can be adapted, reused (in whole or in part), or simply used as guidance.
- Business rules for data requirements.

8.2.3 Output

The objective and output of this step is a conceptual exchange content model that clearly defines business data entities, their semantics, and the relationships among them.

8.3 Map NIEM Components

8.3.1 Description, Content, and Purpose

In this step, an IEPD developer maps NIEM data components to equivalent components defined in the exchange content model. This is generally a very difficult and tedious task and usually requires an understanding of both the exchange content model and NIEM. Even so, finding a complete equivalence mapping from NIEM to an independently developed exchange content model of any significant size is rare. Some domain components will have no equivalent in NIEM; others will have partial equivalents in NIEM; and still others will have complex mappings (many-to-one, one-to-many, or many-to-many relationships).

The lack of an equivalent NIEM component or the inability to cleanly map to exchange content model components may require that the IEPD developer (1) identify such cases in the mapping, and (2) model as NIEM extensions (i.e., create NIEM-NDR-conforming names and structures for) the missing NIEM components per 2.1 (NIEM NDR) in preparation for building XML schemas in the next step. Note that modeling NIEM components in this step is different from Step 2 (Build Exchange content model), where modeling of domain data components (possibly using a local native vocabulary) does not have to conform to the NIEM NDR.

Output from this process is (1) a normative machine-readable artifact defining the current data component map, and (2) a summary report of the mapping. These are specified by:

3.3.2F1 (NIEM Mapping): A normative specification (possibly in XML Schema) that records all details of the semantic links between NIEM components and corresponding components of an exchange content model. This specification requires the identification of:

- Semantic correspondence (in most cases equivalence) between NIEM components and exchange content model components
- NIEM-required metadata and context.
- Structural and semantic differences between the models where they exist.
- Exchange content model components with no equivalent in NIEM.
- NIEM-NDR-conforming extensions corresponding to exchange content model components without equivalent components in NIEM.

3.3.2F2 (NIEM Mapping Report): Specifies a human-readable report of all components that were mapped and the corresponding details, metadata, and explanatory notes (if any), as well as a listing of components that could not be mapped.

Once data requirements that NIEM could not support have been identified and modeled per 2.1 (NIEM NDR), the developer may determine that some of these could be reused by other NIEM users and may be good candidates for new NIEM components. In such cases, the IEPD developer employs 5.2F (CSA Change Request) to request consideration for addition to NIEM. Governance will review all submissions and recommend action to the PMO. Approved components will be integrated into a subsequent NIEM release.

8.3.2 Inputs

This step takes as its input an exchange content model in UML, a derivative of UML, or some other standard data modeling format.

8.3.3 Outputs

- Normative, machine-readable, mapping artifact 3.3.2F1 (NIEM Mapping) that records the correspondence between exchange content model components and NIEM components.
- 3.3.2F2 (NIEM Mapping Report) report of mapping details, metadata, and notes, and nonmapped components.
- NIEM NDR-conforming extension models for components that could not be mapped to exchange content model components.
- (Optional) A NIEM 5.2F (CSA Change Request) that contains the extension models for NIEM NDR-conforming components that the developer believes should be considered for adoption by NIEM.

8.3.4 Tools and Interfaces

Tools and interfaces potentially useful in this step include:

- A NIEM-aware mapping tool that imports an exchange content model in XMI; provides automatic search and mapping capability; identifies potential mappings to NIEM; and suggests appropriate component mappings on the basis of heuristics.
- A NIEM-aware schema subset generation tool that searches NIEM components and metadata; allows user to identify and select components and to generate valid NIEM subset schemas (subsets of a given release reference schema set); and searches all published content (including domain updates and micro, minor, and major releases)
- 5.2S (CSA Change Request SI) system interface for submitting new component requests to the PMO (by proxy through the CSA).

8.4 Build IEPD Artifacts

8.4.1 Description, Content, and Purpose

At this stage, all data and information should be available to build the XML schemas and other associated artifacts for this IEPD. The IEPD developer either builds or generates the IEPD XML schema artifacts defined in 3.1.1 (IEPD Specification). In the course of this step, the IEPD developer may use the NIEM Data Model or any existing IEPDs that can be reused or adapted for this effort.

Once built, NIEM-conforming schema artifacts can be visualized by generating equivalent UML models (as XMI serialized files) from the schemas on the basis of 3.3.1F (NIEM UML Profile).

8.4.2 Inputs

- A release of the NIEM Data Model
- Relevant IEPDs for reuse or adaptation
- BDEM (business context in 1.3F (RefBDEM Export) format)
- 3.3.2F1 (NIEM Mapping), mapping of NIEM components to the exchange content model components.
- (Optional) UML exchange content model per 3.3.1F (NIEM UML Profile) in OMG XMI 2.1 format.
- 5.2F (CSA Change Request) containing new NIEM candidate components.
- (Optional) *wantlist*(s) for reuse, adaptation or migration (per 3.2.1F (NIEM Wantlist)).
- (Optional) local code list per 3.2.2F (IEPD Code List) for generating NIEM-conforming code list schemas.

3.2.1F (NIEM Wantlist): [18] A *wantlist* is an XML schema-specified normative XML listing of NIEM data requirements identified and selected by a user for inclusion in a NIEM schema subset. It does not necessarily include NIEM data components on which those data requirements are dependent (such as parent types, object types, substitution groups, etc.).

3.2.2F (IEPD Code List): [5] A file containing local code values and literals from which a NIEM-conforming code list schema is to be generated. The file should be (1) easy to prepare either manually or by export; and (2) easily parsed or processed by software (e.g. simple spreadsheet, CSV, XML format).

8.4.3 Outputs

The following are specified by 3.1.1 (IEPD Specification):

- XML schemas (XSD):
  - Subset schemas per 2.1 (NIEM NDR).
  - Constraint schemas that are not necessarily NIEM NDR-conforming.
  - Extension schema(s) per 2.1 (NIEM NDR).
  - Exchange schema (i.e., document schema) per 2.1 (NIEM NDR).
  - (Optional) local code list schemas (as needed)
- Wantlist for subset schemas per 3.2.1F (NIEM Wantlist).
- Sample XML Schema instance(s) that validate to IEPD schemas.
- (Optional) XSL stylesheet(s) associated with sample XML instances.
- (Optional) A UML exchange content model based on 3.3.1F (NIEM UML Profile).
- (Optional) An OMG XMI 2.1 serialization of the UML model.

3.3.1F (NIEM UML Profile): Specifies a normative NIEM UML profile that identifies standard UML representations for NIEM, and UML extensions required for NIEM-specific modeling techniques. This profile also employs OMG XMI 2.1.1 [19] for serializing the UML model into XML format that could be reused and processed by other tools.

Although not absolutely necessary, use of NIEM or NIEM-aware capabilities earlier in exchange content modeling and mapping steps can potentially expedite some of the processes leading to IEPD schema production. How this is done will depend on several factors. These may include project requirements and constraints; the modeler's knowledge of UML; the modeler's knowledge of NIEM and the domain; the preexistence of an exchange content model; and other possible factors. There are several ways this could work. If no exchange content model exists, it could be built with NIEM components from the start. This will likely make the mapping step somewhat easier. A more advanced approach would use 3.3.1F (NIEM UML Profile) to build

the exchange content model in UML so that it more formally incorporates NIEM-specific structures. Such a UML model can be used to generate NIEM schemas.

8.4.4 Tools, Interfaces, and References

Tools, interfaces, and references potentially useful in this step include:

- Depending on the methods used by the IEPD developer to build or generate the artifacts, reference 2.1 (NIEM NDR) may be useful.
- A NIEM-aware schema subset generation tool.
- A NIEM-aware mapping tool that, based on exchange content model mapping to NIEM, generates template schemas for IEPD constraint, extension, and exchange schemas.
- A code list schema generator that imports a code list in a format that can be easily prepared by a user or exported from a database and auto-generate associated NIEM-conforming code list schema.
- NIEM-aware migration assistance to translate a NIEM *wantlist* based on one major release forward to a *wantlist* based on the next major release; and to identify difficult translation issues requiring manual intervention and suggest possible fixes.
- Capability to generate a complete sample instance from a schema (or set of schemas.)

3.1.8S (IEPD Schema Gen SI): A system interface that exposes the following functions:

generateSubsetSchema(<Wantlist>)—Input parameter is an instance defined by the 3.2.1F (NIEM Wantlist) format.

generateExtensionSchema(<ExchangeModel>, <MappingArtifact>)—Input parameters are instances defined by (1) 3.3.1F (NIEM UML Profile) represented in the OMG XMI 2.1 format and (2) 3.3.2F1 (NIEM Mapping) formats, respectively.

generateExchangeSchema(<ExchangeModel>, <MappingArtifact>)—Input parameters are instances defined by (1) 3.3.1F (NIEM UML Profile) represented in the OMG XMI 2.1 format and (2) 3.3.2F1 (NIEM Mapping) formats, respectively.

generateConstraintSchema(<ExchangeModel>)—Input parameter is an instance defined by 3.3.1F (NIEM UML Profile) represented in OMG XMI 2.1 format.

generateCodelistSchema(<Spreadsheet>)—Input parameter is an instance defined by 3.2.2F (IEPD Code List) format.

Output for each is an NIEM-conforming schema per 2.1 (NIEM NDR), except for constraint schema output, which is simply W3C XML Schema, not necessarily NIEM-conforming.

8.5 Validate IEPD Artifacts

8.5.1 Description, Content, and Purpose

Major validation tasks include:

- Validate schemas for NIEM NDR conformance—use 2.1 (NIEM NDR) and 2.1F2 (NIEM Conformance Rules) and system interfaces defined below.
- Validate one or more sample XML instances against the schemas (conformance path)—use a standard XML Schema validator (e.g., XMLSpy, Xerces).
- Validate one or more sample XML instances against the constraint schemas (constraint path)—use a standard XML Schema validator (e.g., XMLSpy, Xerces).

All validation errors and issues must be addressed prior to proceeding to the next step. Other types of nonstandard validation and verification should also be performed in this step. Many such tasks currently require manual inspection, some of which could be semiautomated with specialized validation software. Examples include:

- Check general completeness of IEPD schema, instance, and stylesheet artifacts.
- Check for correct use of substitution groups.
- Check that sample instances, stylesheets, and schemas implement as many of the exchange business rules as possible.

8.5.2 Inputs

- 2.1F2 (NIEM Conformance Rules)
- All completed IEPD schemas per 3.1.1 (IEPD Specification)
- Sample XML instances of IEPD per 3.1.1 (IEPD Specification)
- (Optional) Instance display XSL stylesheet(s) per 3.1.1 (IEPD Specification)

8.5.3 Outputs

Outputs include (see 3.1.2S (NIEM Validation SI) below):

- Validation report for schema conformance to NIEM NDR.
- Validation report from an XML validator on sample XML instance(s) and stylesheet(s).
- Validation reports from other validation checks.

8.5.4 Tools, Interfaces, and References

Tools, interfaces, and references potentially useful in this step include:

- As in previous step, depending on the methods used to build or tools used to generate the artifacts, reference 2.1 (NIEM NDR) may be useful.
- Validation system interfaces as follows:

3.1.2S (NIEM Validation SI): A system interface for the following example validation functions (not limited to these):

isSchemaNIEMConformant(<schema(s)>, <NIEM Conformance Rules>)
Input parameters are W3C XML Schema and 2.1F2 (NIEM Conformance Rules) formats, respectively.

checkSubstitutionGroups(<IEPD schemas>, <IEPD Instance>)
Checks integrity and correct use of XSD substitution groups.
Input parameters are XSD schema files and XML instance files, respectively, from an IEPD specified by 3.1.1 (IEPD Specification).

validateXMLInstance(<IEPD schemas>, <IEPD Instance>)
Input parameters are XSD schema files and one XML instance file from an IEPD specified by 3.1.1 (IEPD Specification).

Output from each function is a structured report detailing type and location of each failure; otherwise, the report verifies that schema passes validity check.

- Standard XML Schema validators (e.g., XMLSpy, Xerces).
- Capability to auto-validate NIEM conformance of IEPD schemas by applying NIEM conformance rules.
- Capability to auto-correct specific types of errors identified in IEPD schemas.
- XML editor for correcting artifacts.

8.6 Document and Assemble IEPD Artifacts

8.6.1 Description, Content, and Purpose

At this stage, all IEPD schemas and associated data artifacts (instances and stylesheets) must be complete and valid, and documentation and metadata created or generated in previous steps should be available. The IEPD developer uses 3.1.1 (IEPD Specification) to identify each and determine which is missing or incomplete. Next, the IEPD developer drafts or finalizes required (and appropriate optional) documentation and formats required IEPD metadata for 3.2.3F (IEPD Metadata) file. Next, the developer prepares 3.2.4F (IEPD Catalog) file, an IEPD manifest that records for each file its name, location, purpose, nature, and description. The final subtask is to package all files into an archive file (zip), which is now self-contained, self-described, and capable of being registered into any appropriate repository.

The NIEM TA supports tools that can automatically validate conformance to 3.1.1 (IEPD Specification) including required packaging, files, formats, metadata, etc.

8.6.2 Inputs

Per 3.1.1 (IEPD Specification), the following are collected from all previous steps:

- XML schemas (XSD):
  - Subset schemas
  - Constraint schemas (must conform to W3C XML Schema only; NIEM-conformance is not required)
  - Extension schema(s)
  - Exchange schema (document schema)

- *Wantlist* (XML) for subset schemas per 3.2.1F (NIEM Wantlist).
- Sample XML instance(s) and associated XSL stylesheet(s)
- BDEM business context in 1.3F (RefBDEM Export) format.
- UML exchange content model per 3.3.1F (NIEM UML Profile) in OMG XMI 2.1 format.
- Mapping from exchange content model to NIEM (3.3.2F1 (NIEM Mapping)).
- New NIEM-conforming extension components identified during mapping and modeled to satisfy data requirements in the IEPD that NIEM could not satisfy and that the developer believes may be of significant value to other NIEM users and should be evaluated for integration into a later NIEM release (in format 5.2F (CSA Change Request)).

8.6.3 Outputs

- Other (optional) IEPD documentation and support files as appropriate (in various text and graphic formats).
- 3.2.3F (IEPD Metadata).3.2.4F (IEPD Catalog).
- IEPD archive file containing all above inputs and outputs per 3.1.1 (IEPD Specification).

8.6.4 Tools and Interfaces

Tools and interfaces potentially useful in this step include:

- IEPD Specification-aware validation, assembly, and workspace capability.
- Potential capability: Given a copy of a complete IEPD archive (that includes the IEPD *metadata.xml* file and extension schema), harvest and assemble all data required for a new submission package (per 5.2F (CSA Change Request)).

3.1.4S (IEPD Validation SI): A system interface that accepts an input parameter that is an IEPD archive file per 3.1.1 (IEPD Specification) and executes the following functions:

validateIEPDMetadata(<IEPD archive>)—checks that all required metadata is present

validateIEPDCatalog(<IEPD archive>)—verifies completeness and correctness of catalog

isIEPDComplete(<IEPD archive>)—checks that all required files are present

8.7 Publish IEPD

8.7.1 Description, Content, and Purpose

"Publishing is the process of production and dissemination of literature or information—the activity of making information available for public view. In some cases, authors may be their own publishers." [Reference http://en.wikipedia.org/wiki/Publisher]. For NIEM IEPDs, a

sponsoring organization or other authority may publish, or the author himself/herself may publish, an IEPD. Authors who publish their own IEPDs may decline to submit them for review and certification. However, if authors want recognized approval of their IEPDs, or if NIEM PMO requires an approval process to publish the IEPD to a PMO-sponsored IEPD registry, then PMO must define an IEPD review and certification process (to be described).

This IEPDLC step is complemented by a subscription capability using 3.1.5F (IEPD Subscription). Interested parties may subscribe to receive notifications of newly published IEPDs or updates to existing IEPDs by submitting a 3.1.5F (IEPD Subscription) package. The subscription mechanism could be leveraged for many other NIEM-related notifications, such as releases, NIEM activities, etc.

8.7.2 IEPD Certification and Registration

3.1.6S (IEPD Certification Request SI): A system interface for submitting an IEPD archive per 3.1.1 (IEPD Specification) and 3.1.3F (IEPD Submission) to request PMO certification. This could be as simple as an e-mail with attached 3.1.3F (IEPD Submission) and the IEPD. The IEPD archive file contains all information for evaluation. This interface processes the e-mail by logging the request metadata and storing it with the IEPD archive for subsequent action by the PMO. Steps should include:

Author submits completed IEPD to PMO
PMO reviews, refines, validates, approves, and certifies, or
PMO returns IEPD to source with comments for revision and resubmission

3.1.7S (IEPD Registration SI): A system interface for registering with and distributing to all appropriate registries a PMO-certified IEPD archive. This interface could concurrently notify subscribers of IEPD publication through standard pub/sub mechanisms.

A certification process could be the following sequence (or similar):

- Submit complete 3.1.3F (IEPD Submission) using 3.1.6S (IEPD Certification Request SI).
- Review by PMO or by a governance body on behalf of PMO.
- Approve (or return for refinement and resubmission).
- Prepare and insert official certification document (PDF) into the IEPD archive, and modify *metadata.xml* and *catalog.xhtml* files accordingly per 3.1.1 (IEPD Specification), 3.2.3F (IEPD Metadata) and 3.2.4F (IEPD Catalog).
- Register IEPD archive with and distribute IEPD archive to all appropriate registry/repositories using 3.1.7S (IEPD Registration SI).
- Distribute subscriber notifications based on 3.1.5F (IEPD Subscription) subscription requests.

8.7.3 Inputs

- Uncertified IEPD archive 3.1.1 (IEPD Specification).
- IEPD certification request 3.1.3F (IEPD Submission).

8.7.4 Outputs

- Certified IEPD archive 3.1.1 (IEPD Specification).
- Certified IEPD archive forwarded to appropriate registry/repositories using 3.1.7S (IEPD Registration SI).
- Notifications using 3.1.7S (IEPD Registration SI) to all interested subscribers (who are subscribed via 3.1.5F (IEPD Subscription)).

8.8 Implement IEPD

8.8.1 Description, Content, and Purpose

Given an IEPD specification, a system or application must be built to exchange IEPs (instances) defined by the IEPD. At various times during designing, documenting, implementing, and testing the system, an IEPD Implementation developer will need to perform a number of specific tasks, including but not limited to:

- Identify requirements.
- Search IEPD Implementation Registry for similar or related implementations.
- Reuse, adapt, or borrow ideas from existing IEPD implementations.
- Interface with local databases.
- Map data between local databases and the IEPD.
- Translate or transform data.
- Transfer local data into XML instances to be sent.
- Transfer data in XML instances received into local databases.
- Identify and implement business rules.
- Identify appropriate commercial, government, or open-source software.
- Identify and/or acquire appropriate hardware.
- Adapt, customize, and/or configure software and hardware)

Many major vendors offer tools and/or systems that will perform or assist with these functions and operations (Oracle Application Server 10g and Database 10g, IBM WebSphere, BEA WebLogic, etc.). However, implementations generally still require a relatively large amount of custom work.

8.8.2 Inputs

- Published IEPD archive defined by 3.1.1 (IEPD Specification).
- Other relevant IEPD implementations (4.1F (IEPD Implementation)) that might be reused or adapted for this effort.
- (Optional) NIEM relational model of an IEPD.

8.8.3 Outputs

- Application or system that implements the IEPD.
- APIs, Web Services, or other system interfaces exposed by this system.
- Import/export formats used or required.
- 4.1F (IEPD Implementation) package detailing information about this system.

8.8.4 Tools and Interfaces

Tools and interfaces potentially useful in this step include:

- Existing database ETL (extract, transform, and load) tools can operate on a NIEM relational model of the IEPD. Leveraging such a tool and model can reduce the level of effort and XML/NIEM-specific skills required to implement the IEPD.
- The IEPD search/retrieval specifications identified in this document.
- A NIEM-aware tool that could facilitate building conforming instances.
- Data fusion, link and discovery, and reasoning tools.
- Statistical analysis tools.

9 Data Maturity Model Life Cycle (DMMLC) Perspective

The NIEM Data Maturity Life Cycle (DMMLC) is a reference framework that explains NIEM governance, evolution, optimization, and their associated interactions. The DMMLC involves the NIEM Data Model, the Component Staging Area (CSA), the NIEM Configuration Control Tool (NCCT), and an IEPD Repository. In short, NIEM is evolved and optimized through governance evaluation of (1) feedback and new requirements recorded in NCCT, (2) proposed domain schema updates, and (3) IEPD extension schemas. Approved component changes are integrated into the model. Over time, the new and modified components are incorporated into new and modified IEPDs. The DMMLC is designed to be a self-optimizing, controlled process.

The DMMLC is the overarching process for the evolution of NIEM through a sequence of versions. Details of the DMMLC are defined and specified in the NIEM Version Architecture.

> 5.3G (Version Architecture): [25] A general NIEM specification that defines how changes are identified, applied, controlled, and published as NIEM schemas and within NIEM releases. This document also specifies the processes for various types of releases (e.g., major, minor, micro, domain); and details the CSA, NCCT, related tools and capabilities, and how they assist in governance and evolution of NIEM.

There are two subviews of the DMMLC: Core and domains. These subviews share similarities, yet they are distinct. The domain(s) subview feeds the NIEM Core subview. A description of the purpose, content, processes, and interfaces for each subview follows. Requirements for proposed specifications are identified where appropriate.

9.1 NIEM Core Subview

9.1.1 Description, Content, and Purpose

This subview represents Core change management. All domains depend on Core; therefore, Core updates spawn domain updates that resynchronize to the new Core. NIEM governance bodies evaluate new requirements and feedback in NCCT, IEPD extension schemas, and the most current domain releases. Components from these inputs (and those dependent on them) are uploaded to the component staging area (CSA) for review, modification, harmonization, and vetting. As some changes are approved, they are integrated into NIEM (as vetting continues), and Alphas, Betas, or Release Candidates (RC) are generated as required to support the governance review. Once consensus is reached for all proposed changes, they are integrated into or finalized within NIEM, and a new NIEM release is generated.

The four basic stages to a NIEM (Core and domains) update and the associated operations are:

(1) Identify and prepare content for CSA.

- Search IEPD repositories (or the PMO reference repository), and harvest the components and associated metadata from IEPD extension schemas for evaluation as new candidates for NIEM using 2.3.2S (NIEM Federated Search SI).
- Review NCCT feedback and new data requirements on current NIEM release and, as necessary, extract or prepare new or modified components for evaluation as new candidates as 5.2F (CSA Change Request).
- Prepare (model) and upload 5.2F (CSA Change Request) packages to CSA, possibly using 5.2S (CSA Change Request SI).

(2) Harmonize and vet using 5.1U (CSA WebApp) or 5.1S (CSA SI) to access CSA as a workspace.

- Identify dependencies from and evaluate change impacts to existing IEPDs.
- Identify dependencies from and evaluate change impacts to NIEM itself.
- Harmonize and vet content, semantic overlaps, etc.
- Apply changes to material in CSA as a result of vetting.

(3) Obtain PMO approval.

- Prepare the update to the Core, domain, code lists, etc.
- Apply version rules.
- Validate for conformance (NDR, documentation, etc.).
- Run auto and manual (inspection) quality control checks.
- Submit to PMO.

(4) Integrate into NIEM and generate a new Core release with domains, code lists, etc.

- Model or remodel components as required.
- Generate Alpha, Beta, or Release Candidate (RC) release(s) for review.

The sequence of operations above is notional. Although they appear to be neatly sequenced, in reality many operations are iterative, dependent, and overlapping. For example, during harmonization and vetting, it may be necessary to change material posted to the CSA because of inability to reach consensus, subtle omissions, late additions, etc. Alpha, Beta, and RC domain releases may be generated at various times to support the domain harmonization and vetting stage.

9.1.2 Inputs

- Current NIEM release (including current domain updates released since last NIEM release).
- Current NIEM (model).
- IEPD extension schemas (format: xsd).
- New Core requirements from NCCT prepared as 5.2F (CSA Change Request).
- Core feedback from NCCT prepared as 5.2F (CSA Change Request).
- Semantically overlapping components (across the current NIEM release).

9.1.3 Outputs

- New NIEM release.
- Modified NIEM data model.
- Artifacts for harmonization.
- Dependency reports.
- Change impact reports on NIEM and IEPDs.

9.1.4 Tools and Interfaces

Tools and interfaces potentially useful in this view include:

- Auto identification of dependencies across NIEM.
- Auto-harvest components from extension schemas and domain update submissions.
- NDR-aware modeling tools for new domain components.
- Generate a change impact report.
- Notifications of changes within the CSA.
- Prepare and incorporate components associated with NCCT issues.
- Identify possible semantic overlap among NIEM components.
- Various auto validation capabilities:
    - NDR conformance
    - XML validation
    - Documentation completeness
    - Apply or validate version rules
    - Quality assurance checks

9.2 NIEM Domain Subview

9.2.1 Description, Content, and Purpose

This view represents domain change management, which executes semi-independently of Core. Each domain content model is based on Core but governed by the domain itself under basic NIEM PMO policy and the NDR. Similar to the Core subview, each domain asynchronously evaluates new requirements and feedback in NCCT about its current content model, as well as IEPD extension schemas relevant to its business areas. When the domain reaches consensus, the update is sent to PMO, which reviews the update, and if approved, integrates the content changes and issues a new domain release. Periodically, as specified by 5.3G (Version Architecture), this domain release will be integrated with other domain updates when the Core is updated in the next major NIEM release.

The four basic stages of an exchange content model update and the associated operations are:

(1) Identify and prepare content for CSA.

- Search IEPD repositories (or the PMO reference repository), and harvest the components and associated metadata from IEPD extension schemas relevant to the domain for evaluation as new candidates for NIEM using 5.1U (CSA WebApp) or 5.1S (CSA SI).
- Review NCCT feedback and new data requirements relevant to the current domain release and, as necessary, extract or prepare new or modified components for evaluation as new candidates for NIEM using 5.2F (CSA Change Request).
- Prepare (model) and upload 5.2F (CSA Change Request) packages to CSA, possibly using 5.2S (CSA Change Request SI).

(2) Harmonize and vet using 5.1U (CSA WebApp) or 5.1S (CSA SI) to access CSA as a workspace.

- Identify dependencies from and evaluate change impacts to existing IEPDs.
- Identify dependencies from and evaluate change impacts to other domains.
- Harmonize semantic overlaps with other domains.
- Harmonize and vet content within the domain.
- Coordinate changes to content referenced by other domains.
- Make changes to material in CSA as a result of vetting.

(3) Obtain PMO approval.

- Prepare the domain update.
- Apply version rules.
- Validate for conformance (NDR, documentation, etc.).
- Run auto and manual (inspection) quality control checks.
- Submit to PMO.

(4) Integrate into NIEM and generate a new domain release.

- Model or remodel for promotions to Core.
- Generate Alpha, Beta, or Release Candidate (RC) release(s) for review.

As with the NIEM Core Subview, the notional sequence of operations described above consists of many iterative, dependent, and overlapping operations. The PMO may approve the domain update submission subject to modifications, which might require revetting by the domain or further coordination with dependent domains. Alpha, Beta, and RC domain releases may be generated at various times to support the domain harmonization and vetting stage.

9.2.2 Inputs

- Current domain release.
- Current exchange content model.
- IEPD extension schemas (format: xsd).
- New domain requirements from NCCT prepared as 5.2F (CSA Change Request).
- Domain feedback from NCCT prepared as 5.2F (CSA Change Request).
- Semantically overlapping components (from other domains or Core).
- Components dependent on changes being considered for this domain.

9.2.3 Outputs

- New domain release.
- Modified exchange content model.

Tools or interfaces useful in this view are essentially the same as those for NIEM updates but are domain-specific.

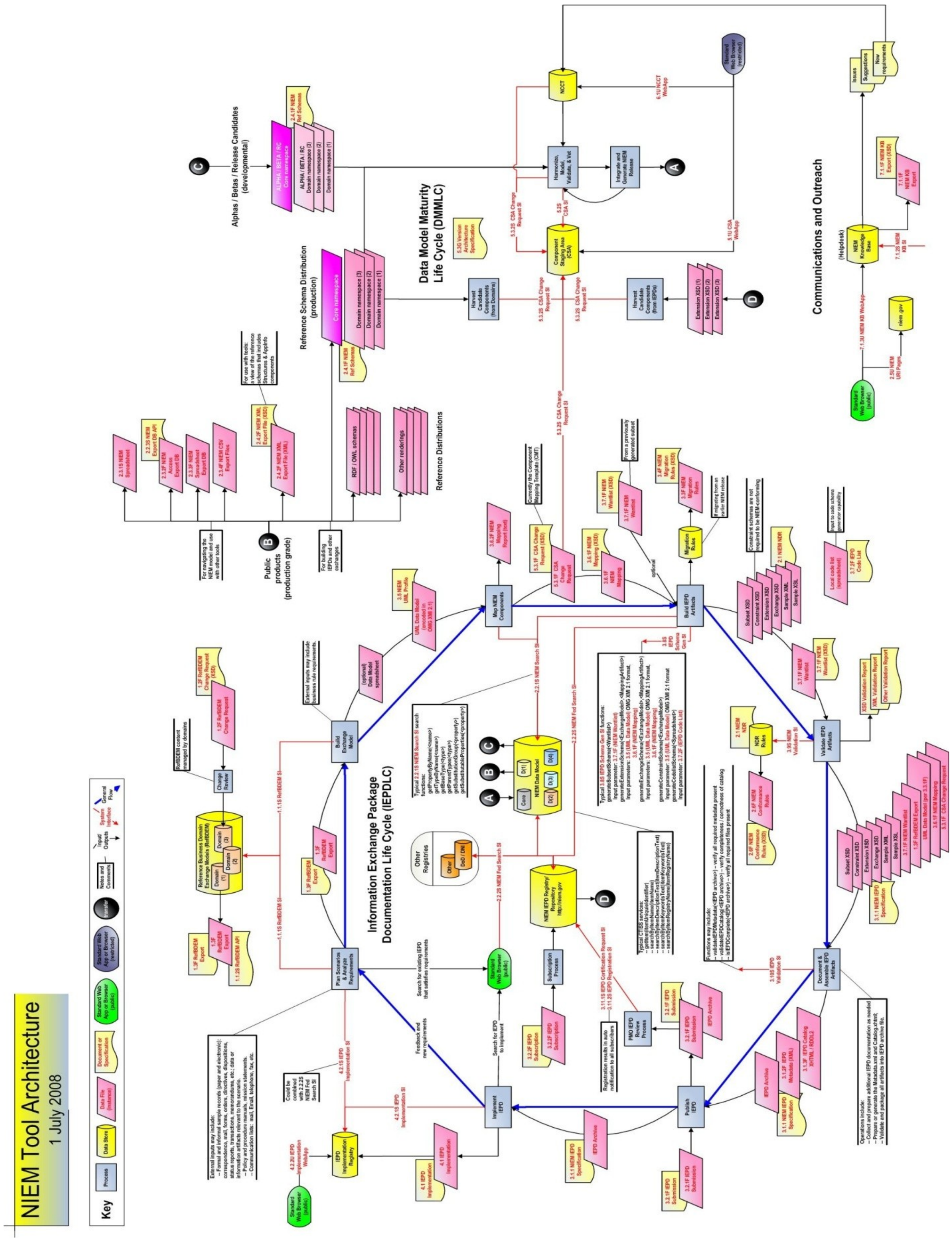
Appendix A: List of Interface Specifications

Specification ID	Description
1.1S1 (RefBDEM SI)	System access to a Reference Business Data Exchange Model (RefBDEM)
1.1S2 (RefBDEM API)	Application Programming Interface (API) or functions for accessing and manipulating a RefBDEM
1.2F (RefBDEM Change Request)	Defines changes requested to a RefBDEM
1.3F (RefBDEM Export)	A RefBDEM database file
2.1 (NIEM NDR)	NIEM Naming and Design Rules (NDR)
2.1F2 (NIEM Conformance Rules)	Formal rules (in Schematron, XSLT, etc.) that represent the NIEM NDR
2.2F1 (NIEM Ref Schemas)	Reference schemas in a NIEM release distribution
2.2F2 (NIEM Spreadsheet)	Microsoft Excel hyperlinked spreadsheet containing NIEM model components for a given release
2.2F3 (NIEM Access Export DB)	Microsoft Access database file containing NIEM model for a given release
2.2F4 (NIEM Spreadsheet Export DB)	Microsoft Excel spreadsheet containing NIEM model for a given release (tabs are database tables)
2.2F5 (NIEM CSV Export Files)	Comma Separated Value (CSV) files of the NIEM model database tables for a given release
2.2F6 (NIEM XML Export File)	XML representation of NIEM release schemas
2.2U (NIEM URI Pages)	Web pages for NIEM components addressable by URI/URL
2.3.1S (NIEM Search SI)	System access to Schema Subset Generation Tool (SSGT) search functionality (NIEM-specific)
2.3.2S (NIEM Federated Search SI)	System access to federated search functionality (generic, not NIEM-specific)

Specification ID	Description
2.3.3S (NIEM Export DB API)	Functions to access/manipulate the NIEM export database
3.1.1 (IEPD Specification)	Specification for content and structure of NIEM Information Exchange Package Documentation (IEPD), a complete self-describing archive file
3.1.2S (NIEM Validation SI)	System interface for validating schemas for NIEM conformance using NIEM conformance rules that represent the NIEM NDR
3.1.3F (IEPD Submission)	File that contains submission metadata and a complete IEPD archive 3.1.3F IEPD Catalog—Defines files in an IEPD archive; manifest for an IEPD (i.e., catalog.xhtml)
3.1.4S (IEPD Validation SI)	System interface for validating IEPD conformance to 3.1.1 IEPD Spec, including metadata and catalog
3.1.5F (IEPD Subscription)	Subscribe to IEPD change notifications
3.1.6S (IEPD Certification Request SI)	System interface for submitting an IEPD and request certification from PMO
3.1.7S (IEPD Registration SI)	System interface to register an IEPD into the PMO IEPD registry
3.1.8S (IEPD Schema Gen SI)	System access to the functionality of a schema generator that produces subset, constraint, extension, and/or exchange schemas
3.2.1F (NIEM Wantlist)	Represents user NIEM data component requirements; import/export to a schema generator that will generate a NIEM subset
3.2.2F (IEPD Code List)	Excel spreadsheet defining one or more code lists: import to a code list schema generator
3.2.3F (IEPD Metadata)	Defines metadata in an IEPD (i.e., metadata.xml)
3.2.4F (IEPD Catalog)	Defines files in an IEPD archive; manifest for an IEPD (i.e., catalog.xhtml)

Specification ID	Description
3.3.1F (NIEM UML Profile)	Defines UML subset used by NIEM Data Model; instance is a UML data model in XMI (may require two different specs)
3.3.2F1 (NIEM Mapping)	A file that formally defines relationships between local data requirements and NIEM data components; import/export to a mapping tool
3.3.2F2 (NIEM Mapping Report)	Summary report of a mapping of local data requirements to NIEM data components
3.4F (NIEM Migration Rules)	Translation rules for converting a Wantlist from one NIEM version to the next
4.1F (IEPD Implementation)	Format for an IEPD implementation package
4.1S (IEPD Implementation SI)	System access to the IEPD Implementation Registry
4.1U (IEPD Implementation WebApp)	Web-based user interface to IEPD Implementation Registry
5.1S (CSA SI)	System access to CSA
5.1U (CSA WebApp)	Web-based user interface to Component Staging Area (CSA)
5.2F (CSA Change Request)	Normative format for NIEM component changes
5.2S (CSA Change Request SI)	System interface for submitting changes to the CSA
5.3G (Version Architecture)	Specifies NIEM change management, change control, and how this is reflected in releases
6.1U (NCCT WebApp)	Web-based user interface to NIEM Configuration Control Tool (NCCT)
7.1F (NIEM KB Export)	Question and answer records in the NIEM Knowledge Base (KB)
7.1S (NIEM KB SI)	System access to the NIEM Knowledge Base
7.1U (NIEM KB WebApp)	Web-based user interface to the NIEM Knowledge Base

Appendix B: NIEM Tool Architecture



Appendix C: NIEM Processes, Operations, Functions

Program Management (PMO)
1. Collect/compute/generate data and statistics (e.g., usage; numbers of properties, types, dependencies; summaries; etc.).
2. Measure performance (of model, IEPDs, training, etc.).
3. Generate artifacts to support PMO decisions and PMO responses to inquiries.
4. Record, track, report, manage, and resolve issues/changes (e.g., NCCT).
5. Facilitate NBAC/NIEM TAC discussions and governance (e.g., SharePoint portal).
6. Support other temporary and permanent committees and tiger teams.
Data Model Maturity Life Cycle (DMMLC)
1. Harmonization (domain to Core, cross-domain, new content).
2. Search and discover NIEM data components.
3. Navigation and visualization of NIEM (the model).
4. Release production and associated products (schemas, documentation, change logs, packaging, etc.).
5. Stage new components as candidates for addition to NIEM.
6. Update and maintain the models for NIEM Core, domains, code lists, etc.
7. Update and track model and component lineage.
8. Support version control, domain independence, code list independence
9. Support requirements traceability.
10. Develop, implement, and stage alpha/beta/RCs associated with a release cycle.
11. Quality assurance, conformance, and consistency checks of model and new content.
12. Generate, process, and/or integrate new content submissions.
13. Check quality, conformance, consistence of Core, domains, etc.
14. Measure various quality aspects of the model (e.g., coupling and cohesion).
15. Assist in submission and wrapping of external standards content.
16. Assist in harmonizing with other external standards.
17. Manage and maintain code lists (aka managed lists).
18. Capabilities to enable domain self-service under the NDR.
19. Exchange content model and namespace management and update.

Data Model Maturity Life Cycle (DMMLC) (continued)
20. Generate reference schemas for domain updates.
21. Domain management
a. Harmonization assistance (e.g., search/identify duplication, semantic overlap).
b. Generation of quality assurance metrics and statistics.
c. Stage new candidate components for addition to a domain.
i. Support for public domain update releases.
ii. Portal support for private domain harmonization and reconciliation.
d. New content submission preparation assistance.
e. Issue resolution.
f. Business Component Layer (BCL) management.
Information Exchange Package Documentation Life Cycle (IEPDLC)
Plan IEPDs:
1. Identify and review data exchange scenarios.
2. Analyze and document data exchange requirements.
3. Build exchange content models (isolates business context from NIEM architecture).
4. Provide common reference business domain exchange models (RefBDEM) to jump-start modeling.
Develop IEPDs:
5. Search and discover NIEM components.
6. Map data components to NIEM.
7. Build or generate subsets, exchange, extension, constraint schemas.
8. Build or generate common NIEM constructs (adapter types, augmentation types, code lists, etc.).
9. Build, assemble, and publish IEPDs.
10. Check IEPD or artifact conformance and consistency.
11. Validate schemas and sample instances for conformance to the NDR.
12. Visualize IEPDs and IEPs (display).
13. Migration assistance (continuous between NIEM versions).

Information Exchange Package Documentation Life Cycle (IEPDLC) (continued)
14. Assistance for using external standards.
15. Facilitate reuse of IEPDs and IEPD artifacts (e.g., to develop new IEPDs).
16. Capture, maintain, and (re)use business context from IEPDs.
17. IEPD Registry / Repository—Publish, search, discover, and share IEPDs.
18. Registry federations for cross-domain or community discovery and reuse.
Implement IEPDs:
19. Implement exchanges based on IEPDs; build a system to exchange IEPs.
20. Integrate with existing local architecture and product suite (many off-the-shelf products exist).
21. Supplement by extension, plug-in, or add-on the most popular software products to facilitate NIEM adoption and time to market.
22. IEPD implementation profile registry; used to publish, search, and discover profiles of IEPD implementations for bootstrapping or streamlining new implementation efforts.
Use IEPDs:
23. Support operational exchange of instances.
24. Optimize throughputs.
25. Document configurations, lessons learned, statistics, measurements, etc.
26. Data and link analysis to discover or infer new data (from instance data).
Help Desk (some are tools themselves):
1. Training (tutorials, presentations, etc.).
2. Web site content (manage, maintain, etc.).
3. Listserv (manage, moderate, etc.).
4. Help desk (logging, management, access, etc.).
5. Knowledge base (management, organization, access, etc.).
6. Technical documentation.
NIEM Communications and Outreach Committee (NCOC)

Appendix D: Tools and Capabilities

These are current existing tools and capabilities accessible from http://niem.gov.

Component Mapping Template (CMT)—NIEM batch content submission format used to build NIEM 1.0 and 2.0. Reference: http://xml.coverpages.org/NIEMv01-20051007-ComponentMappingTemplate.xls
NIEM Mapping Tool—Enables user to upload a UML/XMI data model and map its data components to NIEM component; assists user in selecting appropriate mapping based on heuristics; generates templates for NIEM exchange, extension, and constraint schema based on mapping. Reference: http://niem.gtri.gatech.edu/niemtools/mapping/iN-DEx.iepd
NIEM Component Search—Search and retrieval capability built into SSGT. Reference: http://niem.gtri.gatech.edu/niemtools/ssgt/iN-DEx.iepd
Graphical Browser Tool—A tool that enables visualization of NIEM components and simple relationships to their immediate neighboring components. Reference: http://niem.gtri.gatech.edu/niemtools/ssgt/DataModelViewer.iepd
Schema Subset Generation Tool (SSGT)—Combines search/discovery of NIEM components with schema subset building. Reference: http://niem.gtri.gatech.edu/niemtools/ssgt/iN-DEx.iepd
Schema NDR Conformance and IEPD Validation Tool—Combines NDR and IEPD conformance validation checks (both auto and manual) (not yet publicly available; demonstrated to NIEM TAC in April 2008; Beta testing; will be released when NIEM NDR 1.3 has been approved).
Code List Schema Generation—Enables user to upload a simple Excel spreadsheet of code values (enumerations) for a given NIEM component and generate the NIEM-conforming schema for that component. Reference: http://niem.gtri.gatech.edu/niemtools/build/iN-DEx.iepd
IEPD Support Tool (assemble IEPD with catalog and metadata)—Enables user to enter metadata, arrange files in directories, build catalog file, validate, and assemble NIEM-conforming IEPDs. Reference: http://niem.gtri.gatech.edu/niemtools/iepdt/iN-DEx.iepd
IEPD Conformance validation—Validator for conformance to NIEM IEPD specification; part of the IEPD tool suite. Reference: http://niem.gtri.gatech.edu/niemtools/iepdt/iN-DEx.iepd
IEPD Registry/Repository—A registry and workspace that allows a user to build, validate, store, and share IEPDs; integrated with the SSGT (see above) and Mapping Tool. Reference: http://niem.gtri.gatech.edu/niemtools/iepdt/iN-DEx.iepd

IEPD Clearinghouse—a general IEPD registry (for GJXDM and NIEM). Reference: http://it.ojp.gov/iepd/
Federated Registry—a standards-based registry capability that federates the Defense Information Standards Registry (DISR) and NIEM; built for the PM Information Sharing Environment (PMISE) to expose Common Terrorist Information Sharing Standards (CTISS) (not yet publicly available; Beta testing).
Migration Assistance Tool—Provides IEPD upgrade assistance by forward converting *wantlists* between major sequential versions of NIEM (and GJXDM). Reference: http://niem.gtri.gatech.edu/niemtools/migration/iN-DEx.iepd
NIEM Configuration Control Tool (NCCT)—An open-source, Bugzilla-based tool for collecting, discussing, and deciding issues about NIEM and its associated products. Reference: http://niem.gtri.gatech.edu/ncct/ Reference to Bugzilla: http://www.bugzilla.org
NIEM Knowledge Base—Frequently asked questions (FAQs) and other information logged by the GJXDM/NIEM help desk. Reference: http://gjxdm.custhelp.com/cgi-bin/gjxdm.cfg/php/enduser/std_alp.php?p
Official NIEM Web Site—The official PMO NIEM informational site. Reference: http://niem.gov

Appendix E: Acronyms and Abbreviations

API	Application Programming Interface
BDEM	Business Domain Exchange Model
CMT	Component Mapping Template (or Tool)
CSA	Component Staging Area
CSV	Comma Separated Value (file format)
DB	Database
DHS	U.S. Department of Homeland Security
DMMLC	Data Model Maturity Life Cycle
DOJ	U.S. Department of Justice
ETL	Extract, Transform, and Load
FAQ	Frequently Asked Question
GJXDM	Global Justice XML Data Model
HLTA	High-Level Tool Architecture
IEP	Information Exchange Package
IEPD	Information Exchange Package Documentation
IEPDLC	Information Exchange Package Documentation Life Cycle
KB	Knowledge Base
NCCT	NIEM Configuration Control Tool
NDR	Naming and Design Rules
NIEM	National Information Exchange Model
OJP	U.S. Office of Justice Programs
PMISE	Program Manager Information Sharing Environment
PMO	Program Management Office
RefBDEM	Reference Business Domain Exchange Model

SI	System Interface
TA	Tool Architecture
UML	Universal Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WebApp	Web Application
XMI	XML Metadata Interchange
XML	eXtensible Markup Language
XPath	XML Path Language
XSD	XML Schema Definition
XSL	eXtensible Stylesheet Language
XSLT	eXtensible Stylesheet Language Translation

Appendix F: References

1. Bugzilla project Web site
   http://www.bugzilla.org
2. ebXML Registry Information Model (RIM) v3.0.1, 22 Feb 2007
   http://www.oasis-open.org/committees/download.php/23648/regrep-3.0.1-cd3.zip
3. ebXML Registry Services (RS) v3.0.1, 22 Feb 2007
   http://www.oasis-open.org/committees/download.php/23648/regrep-3.0.1-cd3.zip
4. NIEM Access Export Database
   http://niem.gov/library.php
5. NIEM Code List Template
   http://niem.gtri.gatech.edu/niemtools/resources/template.xls
6. NIEM Concept of Operations v0.5, 9 Jan 2007
   http://niem.gtri.gatech.edu/niemtools/documentation.iepd
7. NIEM CSV Export Database
   http://niem.gov/library.php
8. NIEM IEPD catalog.xhtml specification
   http://niem.gov/topicIN-DEx.php?topic=file-iepdRequirements
9. NIEM IEPD Draft Specification v1.2, Jun 2006
   http://www.niem.gov/topicIN-DEx.php?topic=file-iepdRequirements
10. NIEM IEPD metadata.xml specification
    http://niem.gov/topicIN-DEx.php?topic=file-iepdRequirements
11. NIEM IEPD specification v1.2
    http://niem.gov/topicIN-DEx.php?topic=file-iepdRequirements
12. NIEM Knowledge Base
    http://www.niem.gov/topicIN-DEx.php?topic=link-helpDesk
13. NIEM Naming and Design Rules (NDR) v1.2, 7 Aug 2007
    http://niem.gov/topicIN-DEx.php?topic=file-NDR-lineNum
    (v1.3 in development; working draft available from NIEM TAC SharePoint)
14. NIEM Reference Schemas
    http://niem.gov/library.php
15. NIEM release spreadsheet
    http://niem.gov/library.php
16. NIEM Spreadsheet Export Database
    http://niem.gov/library.php
17. NIEM URI Pages, root reference
    http://niem.gov/niem

18. NIEM Wantlist
http://niem.gtri.gatech.edu/niemtools/resources/schemas/wantlist/niem-1.xsd

19. Object Management Group (OMG), Meta Object Facility/XML Metadata Interchange (XMI) Mapping v2.1.1, Dec 2007
http://www.omg.org/cgi-bin/doc?formal/07-12-02.pdf

20. Quality Assurance Plan and Strategy (QASP) v1.0, May 2008
(to be released on niem.gov soon).

21. Release Strategy
(to be drafted)

22. Resource Directory Description Language (RDDL) v2.0, 16 Oct 2002
http://www.rddl.org/RDDL2

23. Schematron v1.5
http://www.ascc.net/xml/schematron

24. User Guide
(v1.0 in development; current draft available from IJIS)

25. Version Architecture Specification
(v1.0 in development)